2.1 CONFIGURATION

Platform

In this document, the entities to be simulated in a mission-level model are called platforms. Platforms possess various systems such as sensors, communication devices, weapons, and countermeasures; the mission-level model is concerned with the characteristics and performance of these systems and how they interact within a scenario. Within a mission-level model, the operational effectiveness of different platform types is determined by the various system mixes onboard each platform and the system's individual and combined capabilities. The individual platforms are integrated with C3 features and decision-making capabilities. Various sections in this document discuss the individual systems and how they model world reality; C3 and decision-making functions are also discussed at length. However, before the individual systems and their characteristics, capabilities, and functions are discussed, an organizing principle for the platforms must be provided.

Platform capabilities and attributes represent the most detailed level of resolution in a mission-level model. Most decision-making and command and control functions are conducted at the platform level. The functions of systems within the platform are usually directed ultimately at other platforms. Also, perceptions of the simulation's entities is usually held at the platform level.

In SWEG, the organizing principle is the *player-structure*. It is a hierarchy for implementing the various components of a platform and assigning tactical decision-making capabilities, detection susceptibilities, and attack vulnerabilities to the platform. The strength of the *player-structure* is the flexibility it provides the user for organizing the simulation's entities at the appropriate level of detail. A *player* represents what the user thinks is important, and it allows the user to select the amount of implicit and explicit detail in a scenario. The components of a SWEG player are *platforms*, *elements*, *systems*, and *expendables*. These various components have data. It is the definition of data within the *player-structure* that provides a platform with its attributes of configuration, movement, susceptibility, and vulnerability.

Attributes

The platforms in a mission-level model are described in terms of their capabilities and attributes. Differences in operational effectiveness between different platforms are usually the result of different systems and their associated capabilities. These systems are sensors, weapons, communications devices, countermeasures, and decision-making elements; their functional elements are discussed in Sections 2.7 through 2.11. Four attributes – configuration, movement, susceptibility and vulnerability – are also discussed within their own sections.

Various models use different constructs for defining and specifying the capabilities and attributes of a simulation platform. In SWEG this construct is the *player-structure*. While data are usually associated with specific components within the *player-structure*, attributes may be inherited in a downward and/or an upward direction within the *player-structure*. For example, position has both upward and downward inheritance. The *platform* is the *player* component that has position, but a *player* within a *command chain* has implicit knowledge about the positions of the *player*s that are its immediate commander and subordinates. While this knowledge or perception is held about a *player*, the position is actually derived from the positional data for a *platform* within that *player*. For a *player*, positional data is inherited upwards in the *player-structure* hierarchy. On the other hand, during a SWEG weapon-target intercept, the *weapon* system attacks an *element* within the *player*. In order to calculate the intercept point, SWEG needs positional information for the *element*. The position of an *element* is inherited downwards from the parent *platform*.

Movement is another attribute that is inherited in both directions within the *player-structure* hierarchy. A *platform* obtains its ability to move through the definition of a *mover* system within one of its subordinate *element*s. A *platform* can have only one *mover* system. The *element* for which the *mover* system is defined passes movement capability up the *player-structure* hierarchy to its parent *platform*. Since a *platform* can have only one *mover* system, the other *element*s subordinate to the *platform* inherit their movement capabilities from the single *mover* system belonging to the *platform*. The other *system*s defined within the parent *platform*'s *element*s will also move with their parent *element*s and *platform*, and they inherit their movement capability from the *platform*'s single *mover* system.

Vulnerability is an attribute that belongs to the *element* component of the *player-structure*; i.e., the *element* is vulnerable to attack, and the probabilities of kill within SWEG are directed against an *element* and not a *platform* or *player*. The *element* also contains attributes which determine the extent of damage sustained by the *element* following a successful intercept. The criticality of the *element* is one factor that determines the survival or death of the parent *platform* and *player*. (The parent *platform* will also die when its last *element* is destroyed, and the parent *player* will die when its last *platform* is destroyed.)

A *player*'s susceptibility to detection is also determined at the *element* level within the *player-structure* hierarchy. The user-defined susceptibilities contain the *element*'s emittive and reflective signatures in the electromagnetic and acoustic spectra as well as the list of the sensor receivers on other *player*s that are capable of detecting the *element*. By detecting an *element*, the sensor is able to determine the position of the *element* based upon the parent *platform*'s position. In essence, it has detected the *platform*.

Configuration

Platform configuration refers to the physical attributes that have some effect on the movement, susceptibility, vulnerability, and system-based capabilities of the platform. In SWEG, these physical attributes are implemented as the user-defined physical and logical relationships between the components and capabilities of the *player*. The user implements the appropriate configuration of each simulation entity by creating a SWEG *player* to represent the entity and then specifying the *platform*s, *element*s, *system*s, and *expendables* that will implement the entity's attributes and capabilities. The user can structure the *player* type so that all of its components have the same location and, if movement is possible, the same movement path. On the other hand, the structure can be defined with components at several locations, each with separate movement capability. Similarly, the user can specify that components at one location be treated as a single entity or as multiple entities with respect to susceptibility and vulnerability. In addition, a user may arbitrarily select the *systems* (movers, sensors, talkers, thinkers, weapons, or disruptors) and their capabilities and *expendables* at each location.

For example, a surface-to-air-missile (SAM) installation could be modeled as at a single location with a single component to be detected and attacked; or it could be modeled with two separate locations, one with a missile launcher and the other with a radar, operator, and radio. In the first case, all components would survive or be killed together; while in the second case, some capabilities could be retained when other components are killed. The location with multiple systems could be given a single signature and vulnerability, or separate signatures and/or vulnerabilities could be defined for the radar, the radio, and the operator. In fact, the components at a location need not be susceptible or vulnerable at all, and no capabilities need be defined. The user can define components to be at any number greater than or equal to one location, with any number greater than or equal to zero separate susceptibilities, vulnerabilities, and capabilities. The *player* structure directly affects susceptibility, vulnerability, location, and movement, and indirectly affects all other capabilities.

2.1.1 Functional Element Design Requirements

This section contains the design requirements necessary to implement platform configuration in SWEG.

a. SWEG will provide the user with flexibility in defining the structure of a platform implementation so that its configuration produces the user-desired effects on the attributes and capabilities of the simulated platform. SWEG will allow each modeled platform type to have components at one or more locations, with zero or more separately susceptible and/or vulnerable components at each location, and zero or more system capabilities for each separate component at each location. Each system may have zero or more expendables.

b. SWEG will model the following generic types of systems: movers, thinkers, talkers, sensors, weapons, and disruptors. SWEG will provide the capability for the user to specify the capabilities of each subtype of these generic types that will be used in the simulation.

2.1.2 Functional Element Design Approach

SWEG models each platform type with a *player* structure that has five levels of components: *player*, *platform*, *element*, *system* and *expendable*. At each level of the hierarchy, components have data. The following table lists the components, their subordinate components, and the associated data at each level within the player-structure.

TABLE 2.1-1. SWEG Player-Structure.

Component	Subordinate Component	Data
player	platform	perceptions, tactics, orders
platform	element	kinematics, shapes (used for masking)
element	system	status, susceptibilities, vulnerabilities
system	expendable	status, capabilities, comm nets

The SWEG *player-structure* is defined in two places: the TDB and the Scenario Data Base (SDB). Type *players* serve as the blueprints for the scenario *players*. The data that are shared by all *players* of the same type are defined in the TDB; examples of this data include tactics, signatures, and system capabilities. The scenario *player-structure* defines the information specific to each instance of a *player*; this data would include initial position, movement paths, perceptions, orders, and specific frequencies of communication networks. The type *player-structure* facilitates the creation of large scenarios since data that is shared among scenario *players* of the same type need only be defined once. It is the scenario *players* within the SDB that exist and operate in the exercise, and the SDB *player-structure* permits the user to define data specific to each instance of a *player*.

Design Element 1-1: Player

Everything in a SWEG scenario is part of a *player*. In both the TDB and the SDB, the *player* is the top of the *player-structure* hierarchy. In the TDB, tactics are the data that belong to the *player* while the data belonging to a SDB *player* are perceptions and orders.

Tactics, in SWEG terminology, are the mental functions that are common to *players* of the same type; and they include procedures, plans and doctrine. In SWEG, tactics are used for command, control and coordination, weapon employment, information processing, maneuvering, jammer and disruptor employment, communications, and logistics. SWEG mental processes are discussed in extensive detail in the decision-making section of this document.

Perceptions are the knowledge a *player* has about itself and other *players*. Some perceptions are gained implicitly in SWEG. For example, a *player* knows the identity and position of the *players* that are its immediate commander and subordinates on a *command chain*. A *player* can be explicitly given perceptions about other *players*. During the course of a simulation, a *player* updates its perceptions of other *players* through information it directly obtains through its own sensors or information it indirectly obtains through communications with other *players*. Perceptions can be incorrect, incomplete, or out-of-date.

Orders are procedures given to individual scenario *players*. They provide additional refinement for the mental activities and tactics of the type *player*. Orders can include instructions about which tactics should be used in the scenario. They also allow the *player* to implement its tactics in a way that will differentiate the *player* from other *players* with the same tactics. For example, orders can include geographical controls, initial plans, and information related to maneuvers and movement.

Design Element 1-2: Platform

A *player* is required to have at least one *platform*. The functions of the *platform* within the SWEG *player-structure* are to give its parent *player* the capability to be in one or more locations, to move, to be oriented in a direction and rotate that orientation, and to have shape. The movement-related features of a *platform* are termed kinematics in SWEG, and they include the initial position of a *platform*, preplanned paths, and information for dynamically changing paths during the exercise. Type *players* do not have kinematic data; scenario *players'* initial locations and movement paths are defined in the SDB. Movement and kinematics are extensively discussed in Section 2.2 Movement.

In SWEG, *platform*s are point objects by default. However, the user can specify *platform shapes* with combinations of points, lines, areas, and volumes. Shapes allow the user to realistically represent components of a *platform* whose existence is not easily replicated by only points; however, currently these shapes are used only for masking in SWEG. They do not affect signature, vulnerability, movement, etc. The lines, areas, and volumes are not required to touch each other or be connected in any way. If the user defined some number of clouds to be shapes belonging to one *platform*, the clouds could essentially be spread out over the entire exercise and still be part of one *platform*. A shape may be given a type name within the TDB, but it is defined within the SDB.

A *platform* may be defined as *critical* to its parent *player*. If a *critical platform* is destroyed during the exercise, the parent *player* will also be destroyed and removed from the exercise. Criticality is discussed in depth in Section 2.4 Vulnerability.

Design Element 1-3: Element

A *platform* may contain *elements*, but the definition of an *element* within the *player-structure* is optional. A *platform* may have any number of *elements*, and they can be the same or different. *Elements* define a player's susceptibilities to detection by other players and its vulnerabilities to damage and destruction. Susceptibilities define the interactions between the sensors belonging to other *players* and the parent *element*. The parent *element's susceptibility* is defined in terms of its electromagnetic and acoustic *signatures* and the sensor receivers on other *players* which can detect these *signatures*. *Signatures* can be emittive and/or reflective. Signatures are discussed in Section 2.3.

Once an *element* is hit by ordnance from a *weapon*, its predefined quantities and criticality determine the *element's* vulnerability to damage or destruction. An element can be either *discrete* or *continuous*. A *discrete element* can be partially or completely destroyed when attacked; *continuous element*s can be damaged but not destroyed when they are targeted. An *element* can also be defined as *critical* or default to *noncritical*. If one *element* is *critical* to the survival of the *platform*, the death of the *critical element* will cause the death of the *platform* even if other *elements* are left. Quantities and criticality are discussed in Section 2.4 Vulnerability.

The subordination of an *element* to a *platform* ultimately impacts the susceptibility of the *platform* to detection. When an *element* is detected by a sensor receiver, the *element's* location is inherited from its parent *platform*, and the *element* is detected at its parent *platform*'s location. Essentially, the only way to detect a *platform* is to detect one of its *element*s or a *system* on an *element*. If a *platform* has no *elements*, it cannot be detected.

The detection of an *element* by another *player*'s sensors becomes part of the detecting *player*'s perceptions. These perceptions are inputs to the *player*'s decision-making function, and the *player* may decide to transmit an intelligence message containing information of the detected *player*, employ a countermeasure, or launch a weapon as a result of the detection. Which *element*s in a *player* are attacked, the configuration of the *player* and its components ultimately affects its survivability.

Design Element 1-4: Systems

Systems give a player the ability to accomplish tasks: detect others, send messages, think about a contingency plan, or maneuver. An *element* is not required to have *systems*. If *systems* are defined within an *element*, any number of *systems* can be defined and they can be different types. In SWEG there are eight types of *systems*: sensor receivers, sensor transmitters, communications receivers, communications transmitters, weapons, disruptors, movers, and thinkers. All eight types of *systems* have capabilities which are completely user-defined. The actual capabilities will depend upon the type of *system*. *Systems* also have status such as *on*, *off*, or *nonoperational*.

Sensor *systems* (receivers and transmitters) represent a *player's* ability to non-cooperatively obtain information about other *players* in the scenario. Sensors are discussed in Section 2.5. Communication systems allow a *player* to cooperatively share information with another *player*, and they are discussed in Section 2.9. Both sensor and communication systems interact with *players* via energy. Weapon systems allow a *player* to cause physical damage to other *players*. Weapon systems are discussed in Section 2.6. Disruptors give a *player* the ability to interfere with another *player's* attempt to effectively use its sensors, communications and weapons. They are discussed in Section 2.8 Countermeasures.

Mover systems provide the capability to change the location or orientation of a *platform*. A *platform* can have only one mover system. *Mover* systems are different from the other types of SWEG systems; their function does not affect other players. While their purpose is physical in nature, it is independent of other *players*. *Mover* systems can be the agents for allowing any of the interactions caused by the system types already described. Proximity is often a necessary condition for sensing, communicating, shooting, and disrupting. Movement is discussed in Section 2.2.

Thinker systems process information and affect the internal operation of a player. *Thinkers* are agents for the perceptions and decisions. Thinkers may initiate physical processes via these decisions. *Thinkers* have their own data that define their capabilities, but they are also strongly linked to the tactics defined for their parent *player*. If the user gives a *player* the tactics for weapon employment, the user must ensure that at least one *thinker* can use those tactics. *Thinker* systems are discussed in Section 6.1 Decision-making.

Design Element 1-5: Expendables

Expendables are used to regulate the duration for which systems can perform their functions. Ordnance and fuel are examples of expendables. Aircraft on an airbase, prior to their launch, are also expendables for the airbase commander. Items belonging to a *system* can be used up and replenished. SWEG does not require *systems* to have expendables.

2.1.3 Functional Element Software Design

The platform configuration is set up when the user instructions are read; a software approach discussion for this processing is beyond the scope of the current task. No code is used to maintain the platform's structure since it is stored within a dynamic array. The only pertinent code executed during the simulation run affects the creation of new *players* or the components within a *player* and the removal of a *player* from the exercise and the deletion of components within a *player*. The functions of interest are *lifbef* for the creation of new *players* and their components. The functions *immol8*, *injure*, *mutil8*, and *sasin8* are executed to remove a *player* from a scenario and to delete a *player's* components. The *TPlayer::Translate* function is called within *lifbeg*. Since it is a large function in its own right, it is included as a separate code tree.

This section contains one table and seven software code trees which describe the software design necessary to implement the requirements and design approach outlined above. Table

2.1-2 lists most of the functions found in the code trees, and a description of each function is provided. Figure 2.1-1 describes the top level C++ functions in the code for signatures. It contains the path from *main* to all calls to the functions *lifbeg*, *immol8*, *injure*, *mutil8* and *sasin8*. Figure 2.1-2 through 2.1-7 are the detailed code trees for *lifbeg*, *TPlayer::Translate*, *immol8*, *injure*, *mutil8* and *sasin8*.

A function's subtree is provided within the figure only the first time that the function is called. Some functions are extensively called throughout SWEG, and the trees for these functions are in the appendix to this document rather than within each FE description. Within this FE, the functions in that category are *MITRcontrol* and all the member functions in the C++ class *WhereIsIt*.

Not all functions shown in the figures are included in the table. The omitted entries are trivial lookup functions (single assignment statements), list-processing or memory allocation functions, or C++ class functions for construction, etc.

TABLE 2.1-2. Platform Configuration Functions Table.

Function	Description
ailrip	schedules friendly players to notice death from attack
AKSNabsorb	processes reactive actions to absorb someone else
AKSNcontrol	controls resource allocation decisions
AKSNexecute	performs actions to carry out decisions
AKSNrequests	performs actions for filling requests from other players
badpro	processes a randomly failed thinker/processor system
BaseHose::Run	runs all steps
begone	drops a perception block structure for outdated target
BSRVevent	controls sensor physical processing
BSRVonechance	supervises sensor chance calculations
BSRVtrack	determines sensor chance tracking results and effects
cpyiit	copies irregular interval table from TDB to runtime array
CrashTime	determines time of crash
damage	assesses weapon attack results against target
delswt	deletes engaged weapon buffer block from various lists
diefri	deletes friendly perception structures
DLG8control	controls resource evaluation and selection decision
fmtchr	formats characteristics table data for model execution
geocgpts	determines 2-D center of gravity of a set of points
geoctr	determines 2-D center of gravity of a set of points
getlex	formats characteristics table data for model execution
haltit	deletes movement data structures associated with platform
hitbeg	begins the weapon intercept process
immol8	performs self-destruction of platform and possibly parent player
injcom	deletes destroyed communications system

TABLE 2.1-2. Platform Configuration Functions Table. (Contd.)

Function	**Description**
injsnr	deletes destroyed sensor system
injure	deletes systems belonging to an element
itstgt	determines if filter instructions exist for target
jetind	recycles indirect list entries and possibly top list
KILLperception	deletes a sensor-derived perception
lifbeg	begins a new player's life
limevent	eliminates unneeded event nodes data structure
limpendq	eliminates unneeded pending queue data structures
lstaop	searches the allocation option list for the matching option code
main	controls overall execution
MainInit	initiates processing
MainParse	controls parsing of user instructions
MITRcontrol	controls emitter system status changes
MovePlane::CalcAngle	calculates a given angle for orientation
MovePlane::CalcValue	calculates a given value for orientation
MovePlane::GetDuration	gets duration time for a turn
MovePlane::GetRate	gets rate of turn
MovePlane::GetValue	looks up a value for orientation
MovePlane::Set	changes/sets a value
MovePlane::SetDuration	sets duration time of a turn
mutil8	deletes data structures associated with a platform
naybor	controls explicit movement processing
ORNJcontrol	controls construction of the checkpoints of a movement path
prcpot	copies part of an irregular interval table for the RDB
prfmfr	formats miscellaneous values for the RDB
prgeds	generates RDB expendable data structures
prgrcs	generates runtime communications system data structures
prgrjs	generates runtime jammer system data structures
prgrms	generates runtime prime mover system data structures
prgrsh	generates runtime sensor header data structures
prgrsr	generates runtime sensor receiver data base
prgrss	generates runtime sensor system data base
prgrsx	generates runtime sensor transmitter data base
prgrts	generates runtime thinking element data structures
prgrws	generates runtime weapon system data structures
PrintPath	prints a path or a portion of a path
priova	translates SDB antenna orientation data
program	controls execution phase

TABLE 2.1-2. Platform Configuration Functions Table. (Contd.)

Function	Description
prrlus	links a sensor system with other RDB systems
prrnls	gets next linked system pointers
prwkao	formats weapon kill data and ordnance
redwood	adds new entry to or removes top entry from leftist tree
RNDMurn	determines a uniformly distributed random number
sasin8	performs bookkeeping involve with death of a player
semant	controls semantic processing of instructions
sendel	deletes sensor chances
simnxt	controls runtime execution
simphy	controls processing of physical events
simthk	controls processing of mental events
simul8	controls semantic processing of runtime instructions
srhpro	searches table for interval containing a specific value
TAccDirect::AddItem	adds an object pointer to the direct access structure
TAddrData::AddOccupant	adds platform to the list of occupants at this address
TAddrData::AddressVacant	checks address cell for complete vacancy
TAddrData::AddShape	adds shape to the list at this address
TAddrData::DelOccupant	deletes platform from list of occupants at this address
TAddrData::TryDeletion	tries to delete a node in the address tree
TAddress::BottomLevel	determines if address cell is at bottom of tree
TAddress::GetAllCodes	retrieves all address codes including the codes for neighbors
TAddress::GetData	finds the address data object for a specific code
TAddress::GetLevel	finds the level in the address tree corresponding to a specific distance
TAddrNode::GetData	finds the address node for the specific code and adds a new address node if one is not found
TAddrNode::GetNode	returns a pointer to the address data added or found
TEMPchartable	finds a specific characteristics table
TEMPtdbitem	finds a specific TDB item header
TEMPtdbmiscval	stores the value to the address when TDB item header found
TEMPtdbsclr	stores the value to the address when TDB item header found
TExpendable::GasUse	updates fuel usage based on TPathEntry speed and altitude
TExpendable::SearchForType	returns a pointer to expendable matching selected type or a 0 if not found
TMaster::AddPlatform	adds a platform to the direct access list
TMaster::GetCounter	retrieves a user application name counter
TMaster::GetPlatform	retrieves a platform from the direct access list
TMaster::GetPlayer	retrieves a player from the direct access list
TMaster::GetRandomTable	retrieves a random number table

TABLE 2.1-2. Platform Configuration Functions Table. (Contd.)

Function	**Description**
TMaster::GetUanVocab	retrieves a user application name vocabulary entry
TMaster::NewCounter	sets a new user application name counter
TMemory::Allocate	allocates permanent storage
TMemory::AllocTemp	allocates temporary storage
TMemory::Deallocate	deallocates a list of blocks by using the address within the provided pointer
TMemory::DeallocFront	deallocates storage
TMemory::DoAllocate	allocates either permanent or temporary storage
TMemory::LLSTadd2	adds entry to beginning of a list
TMemory::LLSTaddend2	adds entry to end of a list
TMemory::LLSTlength2	counts the number of list entries
TMemory::LLSTlistofflist	searches one list using a second list
TMemory::LLSTmeld	adds entry to list if not already there
TMemory::LLSTremove	returns a pointer to the block on the traversed list which matches the provided key
TMemory::LLSTsearch	searches a list
TMemory::LLSTsearchhard	searches a list using extra parameters
TOrientation::SetFirstTime	sets time of orientation change
TPathEntry::GetBeforePoint	looks up the path point prior to given time
TPathEntry::GetFuelLeft	looks up amount of fuel remaining
TPathEntry::GetPos	looks up position
TPathEntry::GetSpeed	looks up speed
TPathEntry::GetTArrive	looks up arrival time
TPathEntry::GetTDepart	looks up departure time
TPathEntry::GetUnitVel	looks up unit velocity
TPathEntry::PutFuelLeft	stores amount of fuel remaining
TPathEntry::PutUnitVel	stores unit velocity
TPathEntry::RecycleAfterPoint	recycles all points in path after given time
TPlayer::Translate	translates data to set up this player
TTable::RequiredInt	searches a table for a specific integer and returns an error if the integer is not found
TTable::SearchInt	searches a table for a specific integer
TTable::SearchInteger	searches a table for a specific integer
TTerrain::Elevation	returns the z-coordinate of point on surface at a specific location
TTerrain::FindTriangle	determines the terrain triangle for a point given an x, y coordinate pair
typcom	sets up communications type data base
typjam	sets up jammer systems type data base
typmov	sets up type mover data base
typsnr	sets up sensor type data base

TABLE 2.1-2. Platform Configuration Functions Table. (Contd.)

Function	**Description**
typtac	translates tactics data base for player type
typthk	sets up type thinker data base
typwpn	sets up type weapons data base
WhereIsIt::CalcPosition	determines position for a platform given a time
WhereIsIt::CalcUnitVel	determines the unit velocity of a platform given a time
wpnfyr	processes weapon firing effects
wpnhit	controls weapon intercept events
xl8bao	sets up allocation option list
xl8cts	changes TDB and SDB elements into RDB elements
xl8fpl	generates RDB future path list block
xl8gds	generates RDB geographical control
xl8movlimits	sets up movement limits data block
xl8ntp	sets up interrupts
xl8pln	sets up contingency plans
xl8pqs	creates runtime pending queue structures
xl8rao	sets up the resource allocation filter instruction to the input tactic
xl8sdp	initializes sensor-derived perceptions
xl8shape	sets up SDB shape definitions
xl8tad	sets up semantics for RDB orders and directives
yaeail	adds yet another entry to the scenario action item list
yaeail	adds yet another entry to the scenario action item list
yaeres	determines which resource allocation option to add to the pending queue

```
main
    |-BaseHose::Run
        |-MainInit
            |-program
                |-MainParse
                    |-semant
                        |-simul8
                            |-immol8
                            |   |-injure
                            |   |-mutil8
                            |   |-sasin8
                            |-simnxt
                                |-simphy
                                |   |-naybor
                                |   |   |-immol8
                                |   |       |-injure
                                |   |       |-mutil8
                                |   |       |-sasin8
                                |   |-BSRVevent
                                |   |   |-BSRVonechance
                                |   |       |-BSRVtrack
                                |   |           |-immol8
                                |   |               |-injure
                                |   |               |-mutil8
                                |   |               |-sasin8
                                |   |-wpnfyr
                                |   |   |-lifbeg
                                |   |-wpnhit
                                |       |-hitbeg
                                |           |-damage
                                |               |-injure
                                |               |-mutil8
                                |               |-sasin8
                                |-simthk
                                    |-DLG8control
                                        |-AKSNcontrol
                                            |-AKSNexecute
                                                |-AKSNabsorb
                                                |   |-immol8
                                                |       |-injure
                                                |       |-mutil8
                                                |       |-sasin8
                                                |-AKSNrequests
                                                    |-lifbeg
```

FIGURE 2.1-1. Platform Configuration Top-Level Code Tree.

```
lifbeg
        |-TMemory::Index2Ptr
        |-TExpendable::SearchForType
        |-TExpendable::GetDirectory
        |-lifasg
        |       |-DVector::DVector
        |       |-TMemory::Allocate
        |       |-TMemory::Index2Ptr
        |       |-WhereIsIt::CalcPosition
        |       |-DVector::Getx
        |       |-DVector::Gety
        |       \-DVector::Getz
        |-TMemory::Ptr2Index
        |-WhereIsIt::CalcSpeed
        |-lifpth
        |       |-TMemory::Ptr2Index
        |       |-SysToTExpendable
        |       |-TExpendable::SearchForType
        |       |-TExpendable::GetDirectory
        |       |-TMemory::Allocate
        |       |-TMemory::Deallocate
        |       |-WhereIsIt::CalcPosition
        |       |-WhereIsIt::CalcUnitVel
        |       |-DVector::Getx
        |       |-DVector::Gety
        |       |-DVector::Getz
        |       |-WhereIsIt::CalcSpeed
        |       \-TMemory::Index2Ptr
        |-lifnew
        |       |-SysToTExpendable
        |       |-TExpendable::SearchForType
        |       |-TExpendable::GetDirectory
        |       |-TMemory::Index2Ptr
        |       |-WhereIsIt::CalcPosition
        |       |-TPathEntry::SetPos
        |       |-lifdna
        |       |     |-TMemory::Ptr2Index
        |       |     |-TMemory::Index2Ptr
        |       |     |-TPlayer::NewPlayer
        |       |     |     |-TMaster::GetUanVocab
        |       |     |     |-TVocab::get_first_word
        |       |     |     |-TVocab::semcode
        |       |     |     |     |-TVocab::get_hash_table
        |       |     |     |     \-TVocab::str_hash
        |       |     |     |-TMaster::NewCounter
        |       |     |     |-TVocab::add_word
        |       |     |     |     |-TVocab::add_lexcode
        |       |     |     |     |     |-ProgramStop::ProgramStop
        |       |     |     |     |     \-TVocab::new_array
        |       |     |     |     |           |-TMemory::Index2Ptr
        |       |     |     |     |           |-TMemory::Allocate
```

FIGURE 2.1-2. lifbeg Code Tree.

```
|       |    |    |    |         \-TMemory::AllocTemp
|       |    |    |    \-TVocab::add_word_internal
|       |    |    |         |-TVocab::semcode
|       |    |    |         |-TVocab::get_word
|       |    |    |         |    \-TVocab::get_word_header
|       |    |    |         |         |-TVocab::get_hash_table
|       |    |    |         |         \-TVocab::num_hash
|       |    |    |         |-TVocab::new_array
|       |    |    |         |-TVocab::get_hash_table
|       |    |    |         |-TVocab::str_hash
|       |    |    |         \-TVocab::num_hash
|       |    |    |-TMessages::WriteMessage
|       |    |    \-TPlayer::TPlayer
|       |    |         |-TMaster::NewCounter
|       |    |         |-TMaster::AddPlayer
|       |    |         |    |-TAccDirect::TAccDirect
|       |    |         |    |    |-TMemory::Index2Ptr
|       |    |         |    |    \-TMemory::Allocate
|       |    |         |    |-TAccDirect::AddItem
|       |    |         |    |    |-TAccDirect::TAccDirect
|       |    |         |    |    \-TAccDirect::AddItem
|       |    |         |    \-TPlayer::GetID
|       |    |         \-TMemory::Ptr2Index
|       |    |-TPlayer::Getlcdir
|       |    |-TPlayer::Getlsply
|       |    |-TPlayer::GetCmdChains
|       |    |-TMemory::LLSTadd2
|       |    |    |-TMemory::add2
|       |    |    |    \-TMemory::Ptr2Index
|       |    |    |-TMemory::Index2Ptr
|       |    |    |-TMemory::Allocate
|       |    |    \-TMemory::AllocTemp
|       |    |-TMemory::Allocate
|       |    |-TPlayer::Transform
|       |    |    |-TPlayer::Translate
|       |    |    |-typtac
|       |    |    |    |-TMemory::Index2Ptr
|       |    |    |    |-TMemory::Ptr2Index
|       |    |    |    |-TMemory::Allocate
|       |    |    |    |-TEMPtdbitem
|       |    |    |    |    |-TMemory::Index2Ptr
|       |    |    |    |    |-TMemory::LLSTsearch
|       |    |    |    |    |-TVocab::get_first_word
|       |    |    |    |    |-getlex
|       |    |    |    |    |-TVocab::get_word
|       |    |    |    |    |    \-TVocab::get_word
|       |    |    |    |    \-TMessages::WriteMessage
|       |    |    |    |-TMemory::LLSTsearch
|       |    |    |    |-xl8tad
|       |    |    |    |    |-TMemory::Index2Ptr
|       |    |    |    |    |-TMemory::Ptr2Index
|       |    |    |    |    |-TEMPtdbitem
```

FIGURE 2.1-2. lifbeg Code Tree. (Contd.)

```
|     |   |   |   |   |-cpyiit
|     |   |   |   |   |   |-prcpot
|     |   |   |   |   |   |   |-TMemory::Allocate
|     |   |   |   |   |   |   |-TMemory::Index2Ptr
|     |   |   |   |   |   |   \-TTable::RequiredInt
|     |   |   |   |   |   |       |-TTable::SearchInteger
|     |   |   |   |   |   |       |   \-TMemory::Ptr2Index
|     |   |   |   |   |   |       \-TMessages::WriteMessage
|     |   |   |   |   |   \-TMemory::Allocate
|     |   |   |   |   |-TMemory::Allocate
|     |   |   |   |   \-xl8pln
|     |   |   |   |       |-TMemory::Ptr2Index
|     |   |   |   |       |-TMemory::Index2Ptr
|     |   |   |   |       |-TMemory::LLSTsearch
|     |   |   |   |       \-TMemory::LLSTadd2
|     |   |   |   |-TMessages::WriteMessage
|     |   |   |   \-TVocab::get_first_word
|     |   |   |-TMemory::Index2Ptr
|     |   |   |-TMemory::GetLongPtr
|     |   |   |-typsnr
|     |   |   |   |-TMemory::Index2Ptr
|     |   |   |   |-TMemory::Allocate
|     |   |   |   |-fmtchr
|     |   |   |   |   |-TMemory::Ptr2Index
|     |   |   |   |   |-TEMPtdbitem
|     |   |   |   |   |-TMemory::Index2Ptr
|     |   |   |   |   |-TMemory::LLSTlength2
|     |   |   |   |   \-TMemory::Allocate
|     |   |   |   |-prgrsr
|     |   |   |   |   |-TMemory::Ptr2Index
|     |   |   |   |   |-TMemory::Index2Ptr
|     |   |   |   |   |-TMemory::Allocate
|     |   |   |   |   |-TEMPtdbsclr
|     |   |   |   |   |   \-TEMPtdbitem
|     |   |   |   |   |-TEMPtdbitem
|     |   |   |   |   |-TEMPtdbmiscval
|     |   |   |   |   |   \-TMemory::LLSTsearch
|     |   |   |   |   |-TMessages::WriteMessage
|     |   |   |   |   |-cpyiit
|     |   |   |   |   |-fmtchr
|     |   |   |   |   |-TAddress::GetLevel
|     |   |   |   |   |-TVocab::get_first_word
|     |   |   |   |   \-prfmfr
|     |   |   |   |   |   |-TMemory::Ptr2Index
|     |   |   |   |   |   |-TEMPtdbitem
|     |   |   |   |   |   |-TMemory::Index2Ptr
|     |   |   |   |   |   |-TMemory::Allocate
|     |   |   |   |   |   |\-TMemory::LLSTsearch
|     |   |   |   \-prgrsx
|     |   |   |   |   |-TEMPtdbsclr
|     |   |   |   |   |-TEMPtdbitem
|     |   |   |   |   |-TMemory::Index2Ptr
```

FIGURE 2.1-2. lifbeg Code Tree. (Contd.)

```
|   |   |   |   |   |-TMemory::Allocate
|   |   |   |   |   |-TMemory::Ptr2Index
|   |   |   |   |   \-TEMPtdbmiscval
|   |   |   |-typcom
|   |   |   |   |-TMemory::Allocate
|   |   |   |   |-fmtchr
|   |   |   |   |-TMemory::Ptr2Index
|   |   |   |   |-TEMPtdbitem
|   |   |   |   \-TMemory::Index2Ptr
|   |   |   |-typwpn
|   |   |   |   |-TMemory::Index2Ptr
|   |   |   |   |-TMemory::Allocate
|   |   |   |   |-TMemory::Ptr2Index
|   |   |   |   |-TEMPtdbitem
|   |   |   |   |-TMemory::LLSTsearch
|   |   |   |   \-prwkao
|   |   |   |   |-TMemory::Allocate
|   |   |   |   |-TMemory::AllocTemp
|   |   |   |   |-TMemory::Ptr2Index
|   |   |   |   |-TEMPtdbitem
|   |   |   |   |-TMemory::Index2Ptr
|   |   |   |   |-TTable::RequiredInt
|   |   |   |   |-cpyiit
|   |   |   |   \-TMemory::Deallocate
|   |   |   |-typjam
|   |   |   |   |-TMemory::Allocate
|   |   |   |   |-TMemory::Ptr2Index
|   |   |   |   |-TEMPtdbitem
|   |   |   |   |-TMemory::Index2Ptr
|   |   |   |   |-fmtchr
|   |   |   |   |-TMemory::LLSTsearch
|   |   |   |   |-TAddress::GetLevel
|   |   |   |   |-TMessages::WriteMessage
|   |   |   |   \-TVocab::get_first_word
|   |   |   |-typmov
|   |   |   |   |-TMemory::Allocate
|   |   |   |   |-TMemory::Index2Ptr
|   |   |   |   |-TMemory::Ptr2Index
|   |   |   |   |-TEMPtdbitem
|   |   |   |   |-TMemory::LLSTsearch
|   |   |   |   \-prfmfr
|   |   |   \-typthk
|   |   |   |   |-TMemory::Index2Ptr
|   |   |   |   |-TMemory::Allocate
|   |   |   |   |-TMemory::Ptr2Index
|   |   |   |   |-TEMPtdbitem
|   |   |   |   \-TMemory::LLSTlength2
|   |   |-xl8spe
|   |   |   |-TMemory::Index2Ptr
|   |   |   |-TTable::RequiredInt
|   |   |   |-TMaster::GetPlayer
|   |   |   |-TMemory::LLSTadd2
```

FIGURE 2.1-2. lifbeg Code Tree. (Contd.)

```
|      |    |    |-TPlayer::GetInter
|      |    |    |-TMemory::Allocate
|      |    |    |-TMessages::WriteMessage
|      |    |    |-TVocab::get_word
|      |    |    |-xl8fri
|      |    |    |    |-TMemory::Index2Ptr
|      |    |    |    |-TMemory::Ptr2Index
|      |    |    |    |-TMemory::LLSTmeld
|      |    |    |    |-TMemory::Allocate
|      |    |    |    |-TPlayer::GetFirstPlatform
|      |    |    |    |-WhereIsIt::CalcPosition
|      |    |    |    \-DVector::PutVect
|      |    |    |-xl8geo
|      |    |    |    |-TMemory::Index2Ptr
|      |    |    |    |-TMemory::LLSTaddend2
|      |    |    |    |-TMemory::Ptr2Index
|      |    |    |    \-cpypgp
|      |    |    |         |-TMemory::Index2Ptr
|      |    |    |         |-TMemory::LLSTaddend2
|      |    |    |         \-TMemory::Ptr2Index
|      |    |    |-xl8iod
|      |    |    |    |-TMemory::Index2Ptr
|      |    |    |    |-TMemory::Ptr2Index
|      |    |    |    |-TMemory::LLSTaddend2
|      |    |    |    \-xl8omv
|      |    |    |         |-TMemory::Index2Ptr
|      |    |    |         \-TMemory::LLSTaddend2
|      |    |    |-TMemory::Ptr2Index
|      |    |    \-TMemory::LLSTsearch
|      |    |-typtac
|      |    |-xl8rod
|      |    |    |-TMemory::Index2Ptr
|      |    |    |-TMemory::Allocate
|      |    |    |-TMemory::Ptr2Index
|      |    |    |-TEMPtdbitem
|      |    |    |-TMemory::LLSTsearch
|      |    |    |-TMemory::LLSTadd2
|      |    |    |-yaeail
|      |    |    \-TMemory::Deallocate
|      |    |-xl8cns
|      |    |    |-TMemory::Ptr2Index
|      |    |    |-TMemory::LLSTsearch
|      |    |    |-TMessages::WriteMessage
|      |    |    |-TMemory::Allocate
|      |    |    |-xl8ncs
|      |    |    |    |-TMemory::Allocate
|      |    |    |    |-TMemory::Index2Ptr
|      |    |    |    |-TTable::RequiredInt
|      |    |    |    |-TMemory::Ptr2Index
|      |    |    |    |-TMemory::LLSTmeld
|      |    |    |    \-TMemory::LLSTsearch
|      |    |    |-TMemory::LLSTorder
```

FIGURE 2.1-2. lifbeg Code Tree. (Contd.)

```
|     |    |    |    \-TMemory::Index2Ptr
|     |    |    |-TMemory::AllocTemp
|     |    |    |-TMemory::LLSTmeld
|     |    |    |-TMemory::LLSTlength2
|     |    |    |-TMemory::Index2Ptr
|     |    |    \-TMemory::Deallocate
|     |    |-TPlayer::GetActivePlatforms
|     |    \-TAddress::UpdateOtherSensors
|     |         |-TMemory::Index2Ptr
|     |         |-TAddrData::GetParentData
|     |         |    \-TAddrNode::GetParentData
|     |         |-TAddrData::GetPotentialAcqs
|     |         |-ntract
|     |         |    |-TMemory::Index2Ptr
|     |         |    \-TTable::SearchInt
|     |         |-TMemory::Ptr2Index
|     |         \-naysen
|     |              |-TMemory::LLSTsearch
|     |              |-TMemory::Index2Ptr
|     |              |-nayist
|     |              |    |-TMemory::Index2Ptr
|     |              |    |-WhereIsIt::CalcSpeed
|     |              |    |-operator-
|     |              |    |-WhereIsIt::CalcPosition
|     |              |    |-DVector::GetHorizLength
|     |              |    |-isZeroEquiv
|     |              |    \-RNDMurn
|     |              \-inisen
|     |                   |-TMemory::Index2Ptr
|     |                   |-TMemory::Allocate
|     |                   |-TMemory::Ptr2Index
|     |                   |-redwood
|     |                   \-yaeail
|     \-TPlayer::GetFirstPlatform
|-TPlayer::Getlcdir
|-liftel
|     \-TMemory::Index2Ptr
|-lifcnl
|     \-TMemory::Deallocate
|-TMemory::LLSTsearch
|-WhereIsIt::CalcPosition
\-giddyup
      |-TMemory::Index2Ptr
      |-TMemory::Allocate
      |-TMemory::Ptr2Index
      |-DVector::Getx
      |-DVector::Gety
      |-DVector::Putz
      |-TAddress::GetCode
      \-yaeail
```

FIGURE 2.1-2. lifbeg Code Tree. (Contd.)

```
TPlayer::Translate
        |-TMemory::GetLongPtr
        |-TMaster::DebugOn
        |-TMaster::GetUanVocab
        |-TVocab::get_first_word
        |       \-TVocab::get_first_word
        |            \-TVocab::get_hash_table
        |                 \-TMessages::WriteMessage
        |                      |-TMessages::GetMsg
        |                      |-TMessages::PrintALine
        |                      |    \-TSeqFile::Write
        |                      |         \-MFiles::Append
        |                      \-sysdun
        |                           |-TActWindow::GetNext
        |                           |-TActWindow::~TActWindow
        |                           \-ProgramStop::ProgramStop
        |                                \-SwegExcpt::SwegExcpt
        |                                     \-StringDup
        |-TMessages::WriteMessage
        |-TMemory::WriteSummary
        |       |-TMemory::WordsUsed
        |       |-TMemory::CalcWdsLeft
        |       \-CountMemOpns
        |            \-TMaster::DebugOn
        |-TMemory::Allocate
        |-typtac
        |       |-TMemory::Index2Ptr
        |       |-TMemory::Ptr2Index
        |       |-TMemory::Allocate
        |       |-TEMPtdbitem
        |       |    |-TMemory::Index2Ptr
        |       |    |-TMemory::LLSTsearch
        |       |    |    |-TMemory::LLSTsearch
        |       |    |    |    \-TMemory::Index2Ptr
        |       |    |    \-TMemory::Index2Ptr
        |       |    |-TVocab::get_first_word
        |       |    |-getlex
        |       |    |    \-TMessages::WriteMessage
        |       |    |-TVocab::get_word
        |       |    |    \-TVocab::get_word
        |       |    |         \-TVocab::get_word_header
        |       |    |              |-TVocab::get_hash_table
        |       |    |              \-TVocab::num_hash
        |       |    \-TMessages::WriteMessage
        |       |-TMemory::LLSTsearch
        |       |-xl8tad
        |       |    |-TMemory::Index2Ptr
        |       |    |-TMemory::Ptr2Index
        |       |    |-TEMPtdbitem
        |       |    |-cpyiit
        |       |    |    |-prcpot
```

FIGURE 2.1-3. TPlayer::Translate Code Tree.

```
|       |   |   |   |-TMemory::Allocate
|       |   |   |   |-TMemory::Index2Ptr
|       |   |   |   \-TTable::RequiredInt
|       |   |   |       |-TTable::SearchInteger
|       |   |   |       |   \-TMemory::Ptr2Index
|       |   |   |       \-TMessages::WriteMessage
|       |   |   \-TMemory::Allocate
|       |   |-TMemory::Allocate
|       |   \-xl8pln
|       |       |-TMemory::Ptr2Index
|       |       |-TMemory::Index2Ptr
|       |       |-TMemory::LLSTsearch
|       |       \-TMemory::LLSTadd2
|       |           |-TMemory::add2
|       |           |   \-TMemory::Ptr2Index
|       |           |-TMemory::Index2Ptr
|       |           |-TMemory::Allocate
|       |           \-TMemory::AllocTemp
|       |               \-TMemory::DoAllocate
|       |-TMessages::WriteMessage
|       \-TVocab::get_first_word
|-TMemory::Index2Ptr
|-xl8rao
|       |-TMemory::Index2Ptr
|       |-xl8bao
|       |   |-TMemory::Index2Ptr
|       |   |-TMemory::Ptr2Index
|       |   |-TMemory::LLSTmeld
|       |   |   |-TMemory::Index2Ptr
|       |   |   |-TMemory::LLSTsearchhard
|       |   |   |   \-TMemory::Index2Ptr
|       |   |   |-TMemory::Allocate
|       |   |   |-TMemory::AllocTemp
|       |   |   |-TMemory::Ptr2Index
|       |   |   \-TMemory::LLSTaddend2
|       |   |       |-TMemory::Index2Ptr
|       |   |       |-TMemory::Allocate
|       |   |       |-TMemory::AllocTemp
|       |   |       |-TMemory::Ptr2Index
|       |   |       \-TMemory::LLSTsearch
|       |   \-TMemory::LLSTadd2
|       |-lstaop
|       |   |-TMemory::LLSTsearch
|       |   |-TMemory::Ptr2Index
|       |   \-TMemory::Index2Ptr
|       |-TMemory::Allocate
|       \-TMemory::Deallocate
|           |-TMemory::DeallocFront
|           |   \-TMemory::GetBlockLength
|           |-TMemory::Index2Ptr
|           |-CountMemOpns
|           \-TMemory::RcylBlock
```

FIGURE 2.1-3. TPlayer::Translate Code Tree. (Contd.)

```
|             |-TMemory::Index2Ptr
|             \-TMemory::Ptr2Index
|-TMemory::Ptr2Index
|-TEMPtdbitem
|-TMemory::LLSTsearch
|-TMaster::NewCounter
|-TPathEntry::TPathEntry
|     \-DVector::DVector
|-TPathEntry::SetPos
|-WhereIsIt::CalcPosition
|-TPathEntry::SetTDepart
|-TPathEntry::SetTArrive
|-TPathEntry::SetSpeed
|-TPathEntry::GetSpeed
|-TPathEntry::PutUnitVel
|-TPathEntry::GetUnitVel
|-TOrientation::TOrientation
|     |-TMemory::Ptr2Index
|     \-TOrientElement::TOrientElement
|         |-MovePlane::MovePlane
|         |-TMemory::Index2Ptr
|         |-TMemory::Ptr2Index
|         |-TOrientElement::GetNextPtr
|         |   \-TMemory::Index2Ptr
|         \-MovePlane::Set
|             |-MovePlane::CalcValue
|             |   \-MovePlane::CalcAngle
|             |-MovePlane::GetDuration
|             \-MovePlane::SetDuration
|-TOrientation::SetFirstTime
|     |-TMemory::Index2Ptr
|     \-TOrientElement::SetTime
|-WhereIsIt::WhereIsIt
|     \-DVector::DVector
|-TMaster::AddPlatform
|     |-TAccDirect::TAccDirect
|     |   |-TMemory::Index2Ptr
|     |   \-TMemory::Allocate
|     \-TAccDirect::AddItem
|         |-TAccDirect::TAccDirect
|         \-TAccDirect::AddItem
|-TAddress::GetData
|     |-TAddress::GetData
|     |   \-TAddrNode::GetData
|     |       \-TAddrNode::GetNode
|     |           |-TAddrData::GetCode
|     |           |-TAddrData::TAddrData
|     |           |-TAddrNode::TAddrNode
|     |           |   \-MTree::MTree
|     |           \-TAddrData::PutNodePtr
|     \-TAddress::GetCode
|         |-DVector::Getx
```

FIGURE 2.1-3. TPlayer::Translate Code Tree. (Contd.)

```
|           |-DVector::Gety
|           \-TMessages::WriteMessage
|-TAddrData::AddOccupant
|       \-TMemory::Ptr2Index
|-xl8shape
|       |-TMemory::Index2Ptr
|       |-TMemory::Allocate
|       |-TMemory::Ptr2Index
|       |-TEMPtdbitem
|       |-geocgpts
|       |     \-TMemory::Index2Ptr
|       |-TAddress::GetLevel
|       |-dbg_sqrt
|       |-DVector::DVector
|       |-TAddress::GetAllCodes
|       |     |-dbg_pow
|       |     |     \-MathExcpt::MathExcpt
|       |     |           \-SwegExcpt::SwegExcpt
|       |     |-DVector::Getx
|       |     |-DVector::Gety
|       |     \-TMessages::WriteMessage
|       |-TAddress::GetData
|       |-TAddrData::AddShape
|       |     \-TMemory::Ptr2Index
|       \-TEMPchartable
|             |-TEMPtdbitem
|             |-TMemory::LLSTlength2
|             |     \-TMemory::Index2Ptr
|             |-TMemory::Index2Ptr
|             \-TMemory::Allocate
|-xl8cts
|       |-TMaster::DebugOn
|       |-TMessages::WriteMessage
|       |-TMemory::Allocate
|       |-TVocab::get_first_word
|       |-TMemory::WriteSummary
|       |-TMemory::Ptr2Index
|       |-TMemory::LLSTsearch
|       |-xl8gds
|       |     |-TMemory::Ptr2Index
|       |     |-TMaster::DebugOn
|       |     |-TMemory::Index2Ptr
|       |     |-TTable::SearchInt
|       |     |     \-TMemory::Ptr2Index
|       |     |-TMemory::LLSTadd2
|       |     |-TMaster::NewCounter
|       |     |-TMessages::WriteMessage
|       |     |-TVocab::get_first_word
|       |     |-TMemory::WriteSummary
|       |     |-prgeds
|       |     |     |-TMemory::Index2Ptr
|       |     |     |-TMemory::Allocate
```

FIGURE 2.1-3. TPlayer::Translate Code Tree. (Contd.)

```
|     |   |   |-TExpendable::SetType
|     |   |   |-TExpendable::SetCode
|     |   |   |-TExpendable::GetType
|     |   |   |-TExpendable::SetAmount
|     |   |   |-TExpendable::SetNumber
|     |   |   |-TExpendable::SetLocalID
|     |   |   |-TMemory::Ptr2Index
|     |   |   |-TEMPtdbitem
|     |   |   |-TTable::SearchInt
|     |   |   |-TExpendable::SetKind
|     |   |   |-TExpendable::GetKind
|     |   |   |-TExpendable::SetDirectory
|     |   |   |-TExpendable::GetDirectory
|     |   |   |-TExpendable::SetSystem
|     |   |   \-TExpendable::SetNext
|     |   |-prgrss
|     |   |   |-typsnr
|     |   |   |   |-TMemory::Index2Ptr
|     |   |   |   |-TMemory::Allocate
|     |   |   |   |-fmtchr
|     |   |   |   |   |-TMemory::Ptr2Index
|     |   |   |   |   |-TEMPtdbitem
|     |   |   |   |   |-TMemory::Index2Ptr
|     |   |   |   |   |-TMemory::LLSTlength2
|     |   |   |   |   \-TMemory::Allocate
|     |   |   |   |-prgrsr
|     |   |   |   |   |-TMemory::Ptr2Index
|     |   |   |   |   |-TMemory::Index2Ptr
|     |   |   |   |   |-TMemory::Allocate
|     |   |   |   |   |-TEMPtdbsclr
|     |   |   |   |   |   \-TEMPtdbitem
|     |   |   |   |   |-TEMPtdbitem
|     |   |   |   |   |-TEMPtdbmiscval
|     |   |   |   |   |   \-TMemory::LLSTsearch
|     |   |   |   |   |-TMessages::WriteMessage
|     |   |   |   |   |-cpyiit
|     |   |   |   |   |-fmtchr
|     |   |   |   |   |-TAddress::GetLevel
|     |   |   |   |   |-TVocab::get_first_word
|     |   |   |   |   \-prfmfr
|     |   |   |   |       |-TMemory::Ptr2Index
|     |   |   |   |       |-TEMPtdbitem
|     |   |   |   |       |-TMemory::Index2Ptr
|     |   |   |   |       |-TMemory::Allocate
|     |   |   |   |       \-TMemory::LLSTsearch
|     |   |   |   \-prgrsx
|     |   |   |       |-TEMPtdbsclr
|     |   |   |       |-TEMPtdbitem
|     |   |   |       |-TMemory::Index2Ptr
|     |   |   |       |-TMemory::Allocate
|     |   |   |       |-TMemory::Ptr2Index
|     |   |   |       \-TEMPtdbmiscval
```

FIGURE 2.1-3. TPlayer::Translate Code Tree. (Contd.)

```
  |     |   |   |-TMemory::Index2Ptr
  |     |   |   |-TMemory::Allocate
  |     |   |   |-TMemory::Ptr2Index
  |     |   |   |-priova
  |     |   |   |   |-TMemory::Allocate
  |     |   |   |   |-TMemory::Ptr2Index
  |     |   |   |   \-TMemory::LLSTsearch
  |     |   |   |-prgrsh
  |     |   |   |   |-TMemory::LLSTadd2
  |     |   |   |   |-TMemory::Ptr2Index
  |     |   |   |   |-TEMPtdbitem
  |     |   |   |   |-TMemory::LLSTsearch
  |     |   |   |   |-TMemory::Index2Ptr
  |     |   |   |   |-TTable::RequiredInt
  |     |   |   |   |-TMemory::Allocate
  |     |   |   |   |-yaeail
  |     |   |   |   |   |-TMemory::Index2Ptr
  |     |   |   |   |   |-TMaster::GetGameTime
  |     |   |   |   |   |-TMaster::DebugOn
  |     |   |   |   |   |-TMessages::WriteMessage
  |     |   |   |   |   |-TMemory::Allocate
  |     |   |   |   |   |-TMemory::Ptr2Index
  |     |   |   |   |   |-TMaster::PutScenrTree
  |     |   |   |   |   |-redwood
  |     |   |   |   |   |   |-TMemory::Index2Ptr
  |     |   |   |   |   |   \-TMemory::Ptr2Index
  |     |   |   |   |   |-TMaster::GetScenrTree
  |     |   |   |   |   |   \-TMemory::Index2Ptr
  |     |   |   |   |   \-TMaster::GetPhase
  |     |   |   |   \-RNDMurn
  |     |   |   |       \-TMaster::GetRandomTable
  |     |   |   |           \-TMemory::Index2Ptr
  |     |   |   |-TMemory::LLSTadd2
  |     |   |   |-prrlus
  |     |   |   |   |-prrnls
  |     |   |   |   |   |-TMemory::Ptr2Index
  |     |   |   |   |   \-TMemory::LLSTsearch
  |     |   |   |   |-TMemory::Allocate
  |     |   |   |   |-TMemory::Ptr2Index
  |     |   |   |   \-TMemory::LLSTsearch
  |     |   |   \-srhpro
  |     |   |-prgrcs
  |     |   |   |-TMemory::Ptr2Index
  |     |   |   |-typcom
  |     |   |   |   |-TMemory::Allocate
  |     |   |   |   |-fmtchr
  |     |   |   |   |-TMemory::Ptr2Index
  |     |   |   |   |-TEMPtdbitem
  |     |   |   |   \-TMemory::Index2Ptr
  |     |   |   |-TMemory::Allocate
  |     |   |   |-TMemory::Index2Ptr
  |     |   |   |-priova
```

FIGURE 2.1-3. TPlayer::Translate Code Tree. (Contd.)

```
|       |     |     |-prrnls
|       |     |     |-TMemory::LLSTadd2
|       |     |     \-yaeail
|       |     |-prgrws
|       |     |     |-DVector::DVector
|       |     |     |-TMemory::Index2Ptr
|       |     |     |-TMemory::Allocate
|       |     |     |-typwpn
|       |     |     |     |-TMemory::Index2Ptr
|       |     |     |     |-TMemory::Allocate
|       |     |     |     |-TMemory::Ptr2Index
|       |     |     |     |-TEMPtdbitem
|       |     |     |     |-TMemory::LLSTsearch
|       |     |     |     \-prwkao
|       |     |     |           |-TMemory::Allocate
|       |     |     |           |-TMemory::AllocTemp
|       |     |     |           |-TMemory::Ptr2Index
|       |     |     |           |-TEMPtdbitem
|       |     |     |           |-TMemory::Index2Ptr
|       |     |     |           |-TTable::RequiredInt
|       |     |     |           |-cpyiit
|       |     |     |           \-TMemory::Deallocate
|       |     |     |-TMemory::Ptr2Index
|       |     |     |-FloatVector::operator=
|       |     |     |     |-DVector::Getx
|       |     |     |     |-DVector::Gety
|       |     |     |     \-DVector::Getz
|       |     |     |-TEMPtdbitem
|       |     |     |-TMemory::LLSTsearch
|       |     |     |-TTable::RequiredInt
|       |     |     \-prrnls
|       |     |-prgrjs
|       |     |     |-TMemory::Ptr2Index
|       |     |     |-typjam
|       |     |     |     |-TMemory::Allocate
|       |     |     |     |-TMemory::Ptr2Index
|       |     |     |     |-TEMPtdbitem
|       |     |     |     |-TMemory::Index2Ptr
|       |     |     |     |-fmtchr
|       |     |     |     |-TMemory::LLSTsearch
|       |     |     |     |-TAddress::GetLevel
|       |     |     |     |-TMessages::WriteMessage
|       |     |     |     \-TVocab::get_first_word
|       |     |     |-TMemory::Index2Ptr
|       |     |     |-TMemory::Allocate
|       |     |     |-priova
|       |     |     |-TEMPtdbitem
|       |     |     |-TMemory::LLSTsearch
|       |     |     |-TMemory::LLSTadd2
|       |     |     |-isZeroEquiv
|       |     |     |-TMessages::WriteMessage
|       |     |     |-TVocab::get_first_word
```

FIGURE 2.1-3. TPlayer::Translate Code Tree. (Contd.)

```
|       |       |       |-TTable::RequiredInt
|       |       |       \-yaeail
|       |       |-prgrms
|       |       |       |-typmov
|       |       |       |       |-TMemory::Allocate
|       |       |       |       |-TMemory::Index2Ptr
|       |       |       |       |-TMemory::Ptr2Index
|       |       |       |       |-TEMPtdbitem
|       |       |       |       |-TMemory::LLSTsearch
|       |       |       |       \-prfmfr
|       |       |       |-TMemory::Allocate
|       |       |       |-RNDMgrn
|       |       |       |       |-RNDMurn
|       |       |       |       |-dist
|       |       |       |       \-dbg_sqrt
|       |       |       |-TMemory::Index2Ptr
|       |       |       |-TTable::RequiredInt
|       |       |       |-TMemory::Ptr2Index
|       |       |       |-TEMPtdbitem
|       |       |       |-cpyiit
|       |       |       |-SysToTExpendable
|       |       |       |       \-TMemory::Index2Ptr
|       |       |       |-TExpendable::SearchForType
|       |       |       |       |-TExpendable::GetNextPtr
|       |       |       |       |       \-TMemory::Index2Ptr
|       |       |       |       \-TExpendable::GetType
|       |       |       \-TMessages::WriteMessage
|       |       |-prgrts
|       |       |       |-typthk
|       |       |       |       |-TMemory::Index2Ptr
|       |       |       |       |-TMemory::Allocate
|       |       |       |       |-TMemory::Ptr2Index
|       |       |       |       |-TEMPtdbitem
|       |       |       |       \-TMemory::LLSTlength2
|       |       |       |-TMemory::Index2Ptr
|       |       |       |-TMemory::Allocate
|       |       |       |-TMemory::Ptr2Index
|       |       |       |-TMemory::LLSTmeld
|       |       |       |-prrnls
|       |       |       |-TMemory::LLSTsearch
|       |       |       \-TTable::RequiredInt
|       |       |-TEMPchartable
|       |       |-TMemory::Allocate
|       |       \-TEMPtdbitem
|       |-TMemory::Index2Ptr
|       \-TMaster::NewCounter
|-TMemory::AllocTemp
|-TPlayer::GetActivePlatforms
|       \-TMemory::Index2Ptr
|-TMemory::Deallocate
|-xl8pqs
|       |-TMemory::LLSTlength2
```

FIGURE 2.1-3. TPlayer::Translate Code Tree. (Contd.)

```
|       |-TMemory::Index2Ptr
|       |-TMemory::Allocate
|       |-TMemory::Deallocate
|       |-TEMPtdbitem
|       \-xl8ntp
|            |-TMemory::Index2Ptr
|            |-TMemory::LLSTadd2
|            \-TMemory::Ptr2Index
|-xl8sdp
|       |-TMemory::Index2Ptr
|       |-lstaop
|       |-TMemory::Allocate
|       |-TMemory::Ptr2Index
|       |-TMaster::GetPlayer
|       |    |-TMaster::GetCounter
|       |    |-TMaster::GetPlayer
|       |    |    \-TAccDirect::GetItem
|       |    \-TPlayer::Matches
|       |-TPlayer::GetInter
|       |-TPlayer::GetActivePlatforms
|       |-TMemory::LLSTlength2
|       |-yaeres
|       |    |-TMemory::Index2Ptr
|       |    |-itstgt
|       |    |    |-TMemory::LLSTsearch
|       |    |    \-TMemory::LLSTsearch
|       |    |-TMemory::Allocate
|       |    |-yaeail
|       |    \-TMemory::Ptr2Index
|       |-TMessages::WriteMessage
|       \-TVocab::get_first_word
|-xl8fpl
|       |-TMaster::GetTerrain
|       |-TMemory::LLSTlistofflist
|       |    |-TMemory::Index2Ptr
|       |    |-TMemory::LLSTsearchhard
|       |    |    |-TMemory::LLSTsearchhard
|       |    |    \-TMemory::Index2Ptr
|       |    \-TMemory::Ptr2Index
|       |-TMemory::Index2Ptr
|       |-TMemory::Allocate
|       |-TMemory::Ptr2Index
|       |-xl8movlimits
|       |    |-TMemory::Index2Ptr
|       |    |-TMemory::Allocate
|       |    \-TMemory::Ptr2Index
|       |-geoctr
|       |    \-dbg_sqrt
|       |-FloatVector::Getz
|       |-DVector::DVector
|       |    |-FloatVector::Getx
|       |    |-FloatVector::Gety
```

FIGURE 2.1-3. TPlayer::Translate Code Tree. (Contd.)

```
|       |     \-FloatVector::Getz
|       |-TMaster::TerrainOn
|       |-TTerrain::Elevation
|       |     |-DVector::Getx
|       |     |-DVector::Gety
|       |     |-VertexIndex::VertexIndex
|       |     |     \-VertexIndex::operator=
|       |     |-TTerrain::FindTriangle
|       |     |     |-VertexIndex::VertexIndex
|       |     |     |-TAddress::Cartesian2Spherical
|       |     |     |     |-dbg_sqrt
|       |     |     |     |-dbg_asin
|       |     |     |     |     |-MathExcpt::MathExcpt
|       |     |     |     |     \-asin_c
|       |     |     |     \-dbg_atan2
|       |     |     |-TMessages::WriteMessage
|       |     |     |-TAddress::Spherical2Cartesian
|       |     |     |     \-TMaster::DebugOn
|       |     |     |-VertexIndex::operator=
|       |     |     |-VerticeArray::operator[]
|       |     |     |     \-VertexIndex::Value
|       |     |     |-TTerrain::toPtr
|       |     |     |     \-SwegExcpt::SwegExcpt
|       |     |     |-operator-
|       |     |     |     \-VertexIndex::VertexIndex
|       |     |     |-VertexIndex::operator++
|       |     |     |-dist
|       |     |     |-operator+
|       |     |     |     \-VertexIndex::VertexIndex
|       |     |     \-VertexIndex::operator+=
|       |     |           \-operator+
|       |     |-VerticeArray::operator[]
|       |     |-operator+
|       |     \-isZeroEquiv
|       |-FloatVector::Putz
|       |-DVector::Getz
|       |-FloatVector::operator=
|       |-operator+
|       |-TOrientation::SetFirstTime
|       |-ORNJcontrol
|       |-TPathEntry::RecycleAfterPoint
|       |     |-TPathEntry::SetNext
|       |     |-TMemory::Deallocate
|       |     \-TPathEntry::GetNext
|       |-TPathEntry::~TPathEntry
|       |-SysToTExpendable
|       |-TExpendable::GasUse
|       |     |-TMemory::Index2Ptr
|       |     |-TExpendable::GetSystem
|       |     |-isZeroEquiv
|       |     |-TPathEntry::GetFuelLeft
|       |     |-TPathEntry::PutFuelLeft
```

FIGURE 2.1-3. TPlayer::Translate Code Tree. (Contd.)

```
|       |     |-TExpendable::GetAmount
|       |     |-TPathEntry::GetNextPtr
|       |     \-TPathEntry::GetTDepart
|       |-TMaster::DebugOn
|       |-PrintPath
|       |     |-TMaster::GetTerrain
|       |     |-chrchr
|       |     |-TMaster::TerrainOn
|       |     |-TPathEntry::GetSpeed
|       |     |-TPathEntry::GetTDepart
|       |     |-TPathEntry::GetPos
|       |     |-TPathEntry::GetTArrive
|       |     |-DVector::Getz
|       |     |-TTerrain::Elevation
|       |     |-TPathEntry::GetUnitVel
|       |     |-isZeroEquiv
|       |     |-DVector::Getx
|       |     |-DVector::Gety
|       |     |-dbg_atan2
|       |     |-dbg_asin
|       |     |-numtim
|       |     |-TPathEntry::GetFuelLeft
|       |     |-TPathEntry::GetNextPtr
|       |     \-TOrientation::PrintOrientations
|       |           |-TMemory::Index2Ptr
|       |           |-TOrientElement::GetNext
|       |           |-TOrientElement::GetTime
|       |           \-TOrientElement::Print
|       |                 |-numtim
|       |                 |-MovePlane::GetValue
|       |                 |-MovePlane::GetRate
|       |                 \-MovePlane::GetDuration
|       |-CrashTime
|       |     |-TMaster::GetTerrain
|       |     |-TMaster::TerrainOn
|       |     |-TPathEntry::GetPos
|       |     |-DVector::Getz
|       |     |-TTerrain::Elevation
|       |     |-TPathEntry::GetNextPtr
|       |     |-WhereIsIt::CalcUnitVel
|       |     |-TPathEntry::GetTDepart
|       |     \-TPathEntry::GetSpeed
|       \-yaeail
\-TMemory::Deallocate
```

FIGURE 2.1-3. TPlayer::Translate Code Tree. (Contd.)

```
immol8
      |-injure
      |      |-TMemory::Index2Ptr
      |      |-injsnr
      |      |    |-TMemory::Index2Ptr
      |      |    |-TMemory::Ptr2Index
      |      |    |-MITRcontrol
      |      |    |-jetind
      |      |    |    |-TMemory::LLSTremove
      |      |    |    \-TMemory::Deallocate
      |      |    |-TMemory::Deallocate
      |      |    \-TMemory::LLSTremove
      |      |-injcom
      |      |    |-TMemory::Index2Ptr
      |      |    |-MITRcontrol
      |      |    |-TMemory::Deallocate
      |      |    |-jetind
      |      |    \-TMemory::LLSTremove
      |      |-delswt
      |      |-MITRcontrol
      |      |-jetind
      |      |-TMemory::Deallocate
      |      |-TMemory::LLSTremove
      |      \-badpro
      |           |-TMemory::Index2Ptr
      |           |-redwood
      |           |-TMemory::LLSTremove
      |           |-limpendq
      |           |    \-TMemory::Deallocate
      |           |-TMemory::Deallocate
      |           |-TMemory::Ptr2Index
      |           \-limevent
      |-mutil8
      |      |-TMemory::Index2Ptr
      |      |-haltit
      |      |    |-TMemory::Index2Ptr
      |      |    |-TMemory::Ptr2Index
      |      |    |-TMemory::Deallocate
      |      |    |-TPathEntry::GetBeforePoint
      |      |    \-TPathEntry::SetTDepart
      |      |-TMemory::LLSTremove
      |      |-TAddrData::DelOccupant
      |      |    |-TMemory::LLSTremove
      |      |    \-TAddrData::TryDeletion
      |      |-TMemory::LLSTsearch
      |      |-TMemory::Ptr2Index
      |      |-limevent
      |      |-yaeail
      |      |-TMemory::Deallocate
      |      |-TMemory::LLSTlistofflist
      |      \-TMaster::GetPlatform
      |
```

FIGURE 2.1-4. immol8 Code Tree.

```
          |-TMemory::Ptr2Index
          |-ailrip
          |        |-TMemory::Index2Ptr
          |        |-TMaster::GetPlayer
          |        |-TPlayer::IsAlive
          |        |-TMemory::Allocate
          |        |-TMemory::Ptr2Index
          |        \-yaeail
          |-TMemory::Index2Ptr
          \-sasin8
                   |-TMemory::Index2Ptr
                   |-TMaster::GetEnvironment
                   |     \-TMemory::Index2Ptr
                   |-TMemory::Ptr2Index
                   |-diefri
                   |     |-TMemory::Ptr2Index
                   |     |-TMemory::LLSTsearch
                   |     |-TMemory::Deallocate
                   |     |-TMemory::LLSTremove
                   |     |-jetind
                   |     \-TMemory::Index2Ptr
                   |-KILLperception
                   |     |-TMemory::Ptr2Index
                   |     |-TMemory::Index2Ptr
                   |     \-begone
                   |           |-TMemory::LLSTremove
                   |           |-TMemory::Deallocate
                   |           |-delswt
                   |           |-MITRcontrol
                   |           \-TMemory::Index2Ptr
                   |-TMemory::Deallocate
                   |-TMemory::Deallocate
                   |-limevent
                   |-TMemory::LLSTremove
                   |-redwood
                   \-limpendq
```

FIGURE 2.1-4. immol8 Code Tree. (Contd.)

```
 injure
          |-TMemory::Index2Ptr
          |-injsnr
          |        |-TMemory::Index2Ptr
          |        |-TMemory::Ptr2Index
          |        |-MITRcontrol
          |        |-jetind
          |        |     |-TMemory::LLSTremove
          |        |     \-TMemory::Deallocate
          |        |-TMemory::Deallocate
          |        \-TMemory::LLSTremove
```

FIGURE 2.1-5. injure Code Tree.

```
    |-injcom
    |       |-TMemory::Index2Ptr
    |       |-MITRcontrol
    |       |-TMemory::Deallocate
    |       |-jetind
    |       \-TMemory::LLSTremove
    |-delswt
    |-MITRcontrol
    |-jetind
    |-TMemory::Deallocate
    |-TMemory::LLSTremove
    \-badpro
            |-TMemory::Index2Ptr
            |-redwood
            |-TMemory::LLSTremove
            |-limpendq
            |    \-TMemory::Deallocate
            |-TMemory::Deallocate
            |-TMemory::Ptr2Index
            \- limevent
```

FIGURE 2.1-5. injure Code Tree. (Contd.)

```
mutil8
    |-TMemory::Index2Ptr
    |-haltit
    |       |-TMemory::Index2Ptr
    |       |-TMemory::Ptr2Index
    |       |-TMemory::Deallocate
    |       |    |-TMemory::DeallocFront
    |       |    |    \-TMemory::GetBlockLength
    |       |    |-TMemory::Index2Ptr
    |       |    |-CountMemOpns
    |       |    |    \-TMaster::DebugOn
    |       |    \-TMemory::RcylBlock
    |       |         |-TMemory::Index2Ptr
    |       |         \-TMemory::Ptr2Index
    |       |-TPathEntry::GetBeforePoint
    |       |    |-TMemory::Index2Ptr
    |       |    |-TPathEntry::GetTArrive
    |       |    |-TPathEntry::GetTDepart
    |       |    |-TPathEntry::GetNext
    |       |    |-TPathEntry::GetPrev
    |       |    \-TMemory::Ptr2Index
    |       \-TPathEntry::SetTDepart
    |-TMemory::LLSTremove
    |       |-TMemory::Index2Ptr
    |       |-TMemory::LLSTsearch
    |       |    \-TMemory::Index2Ptr
    |       \-TMemory::Deallocate
    |-TAddrData::DelOccupant
    |       |-TMemory::LLSTremove
```

FIGURE 2.1-6. mutil8 Code Tree.

```
|       \-TAddrData::TryDeletion
|           |-TAddrData::AddressVacant
|           |-TAddress::BottomLevel
|           |-MTree::DeleteNode
|           |    \-LLSTremovePtr
|           |         |-LLSTsearchPtr
|           |         \-TMemory::Deallocate
|           |-TAddrNode::~TAddrNode
|           \-TAddrData::~TAddrData
|-TMemory::LLSTsearch
|       |-TMemory::LLSTsearch
|       |    \-TMemory::Index2Ptr
|       \-TMemory::Index2Ptr
|-TMemory::Ptr2Index
|-limevent
|       |-TMemory::Deallocate
|       |-TMemory::Index2Ptr
|       |-TMaster::GetPlayer
|       |    \-TAccDirect::GetItem
|       |         \-TAccDirect::GetItem
|       |-TPlayer::IsAlive
|       |-TMemory::LLSTsearch
|       \-TMemory::LLSTremove
|-yaeail
|       |-TMemory::Index2Ptr
|       |-TMaster::GetGameTime
|       |-TMaster::DebugOn
|       |-TMessages::WriteMessage
|       |    |-TMessages::GetMsg
|       |    |-TMessages::PrintALine
|       |    |    \-TSeqFile::Write
|       |    |         \-MFiles::Append
|       |    \-sysdun
|       |         |-TActWindow::GetNext
|       |         |-TActWindow::~TActWindow
|       |         \-ProgramStop::ProgramStop
|       |              \-SwegExcpt::SwegExcpt
|       |                   \-StringDup
|       |-TMemory::Allocate
|       |    \-TMemory::DoAllocate
|       |-TMemory::Ptr2Index
|       |-TMaster::PutScenrTree
|       |-redwood
|       |    |-TMemory::Index2Ptr
|       |    \-TMemory::Ptr2Index
|       |-TMaster::GetScenrTree
|       |    \-TMemory::Index2Ptr
|       \-TMaster::GetPhase
|-TMemory::Deallocate
|-TMemory::LLSTlistofflist
|       |-TMemory::Index2Ptr
|       |-TMemory::LLSTsearchhard
```

FIGURE 2.1-6. mutil8 Code Tree. (Contd.)

```
        |       |    |-TMemory::LLSTsearchhard
        |       |    |    \-TMemory::Index2Ptr
        |       |    \-TMemory::Index2Ptr
        |       \-TMemory::Ptr2Index
        \-TMaster::GetPlatform
```

FIGURE 2.1-6. mutil8 Code Tree. (Contd.)

```
sasin8
        |-TMemory::Index2Ptr
        |-TMaster::GetEnvironment
        |       \-TMemory::Index2Ptr
        |-TMemory::Ptr2Index
        |-diefri
        |       |-TMemory::Ptr2Index
        |       |-TMemory::LLSTsearch
        |       |    |-TMemory::LLSTsearch
        |       |    |    \-TMemory::Index2Ptr
        |       |    \-TMemory::Index2Ptr
        |       |-TMemory::Deallocate
        |       |-TMemory::LLSTremove
        |       |    |-TMemory::Index2Ptr
        |       |    |-TMemory::LLSTsearch
        |       |    |    \-TMemory::Index2Ptr
        |       |    \-TMemory::Deallocate
        |       |-jetind
        |       |    |-TMemory::LLSTremove
        |       |    \-TMemory::Deallocate
        |       \-TMemory::Index2Ptr
        |-KILLperception
        |       |-TMemory::Ptr2Index
        |       |-TMemory::Index2Ptr
        |       \-begone
        |            |-TMemory::LLSTremove
        |            |-TMemory::Deallocate
        |            |-delswt
        |            |    |-TMemory::LLSTremove
        |            |    |-TMemory::Index2Ptr
        |            |    |-TMemory::LLSTlistofflist
        |            |    |    |-TMemory::Index2Ptr
        |            |    |    |-TMemory::LLSTsearchhard
        |            |    |    |    |-TMemory::LLSTsearchhard
        |            |    |    |    |    \-TMemory::Index2Ptr
        |            |    |    |    \-TMemory::Index2Ptr
        |            |    |    \-TMemory::Ptr2Index
        |            |    |-TMaster::GetPlayer
        |            |    |    \-TAccDirect::GetItem
        |            |    |-TPlayer::IsAlive
        |            |    |-TMemory::Ptr2Index
        |            |    |-TMemory::LLSTsearch
```

FIGURE 2.1-7. sasin8 Code Tree.

```
|     |   |-limevent
|     |   |   |-TMemory::Deallocate
|     |   |   |-TMemory::Index2Ptr
|     |   |   |-TMaster::GetPlayer
|     |   |   |-TPlayer::IsAlive
|     |   |   |-TMemory::LLSTsearch
|     |   |   \-TMemory::LLSTremove
|     |   |-TMemory::Allocate
|     |   |   \-TMemory::DoAllocate
|     |   |-yaeail
|     |   |   |-TMemory::Index2Ptr
|     |   |   |-TMaster::GetGameTime
|     |   |   |-TMaster::DebugOn
|     |   |   |-TMessages::WriteMessage
|     |   |   |-TMemory::Allocate
|     |   |   |-TMemory::Ptr2Index
|     |   |   |-TMaster::PutScenrTree
|     |   |   |-redwood
|     |   |   |   |-TMemory::Index2Ptr
|     |   |   |   \-TMemory::Ptr2Index
|     |   |   |-TMaster::GetScenrTree
|     |   |   |   \-TMemory::Index2Ptr
|     |   |   |-TMaster::GetPhase
|     |   |-sendel
|     |   |   |-TMemory::LLSTremove
|     |   |   |-TMemory::Ptr2Index
|     |   |   |-TMemory::Deallocate
|     |   |   |-TMemory::Index2Ptr
|     |   |   |-limevent
|     |   |   |-redwood
|     |   |   |-TMemory::Deallocate
|     |   |   |-TMemory::Allocate
|     |   |   \-yaeail
|     |   |-TMaster::GetPlatform
|     |   \-TMemory::Deallocate
|     |-MITRcontrol
|     \-TMemory::Index2Ptr
|-TMemory::Deallocate
|-TMemory::Deallocate
|-limevent
|-TMemory::LLSTremove
|-redwood
\-limpendq
    \-TMemory::Deallocate
```

FIGURE 2.1-7. sasin8 Code Tree. (Contd.)

2.1.4 Assumptions and Limitations

Aggregation changes are limited to changing an expendable into a PLAYER with a single PLATFORM or changing a PLATFORM into an expendable item belonging to a thinker.

2.1.5 Known Problems or Anomalies

None.